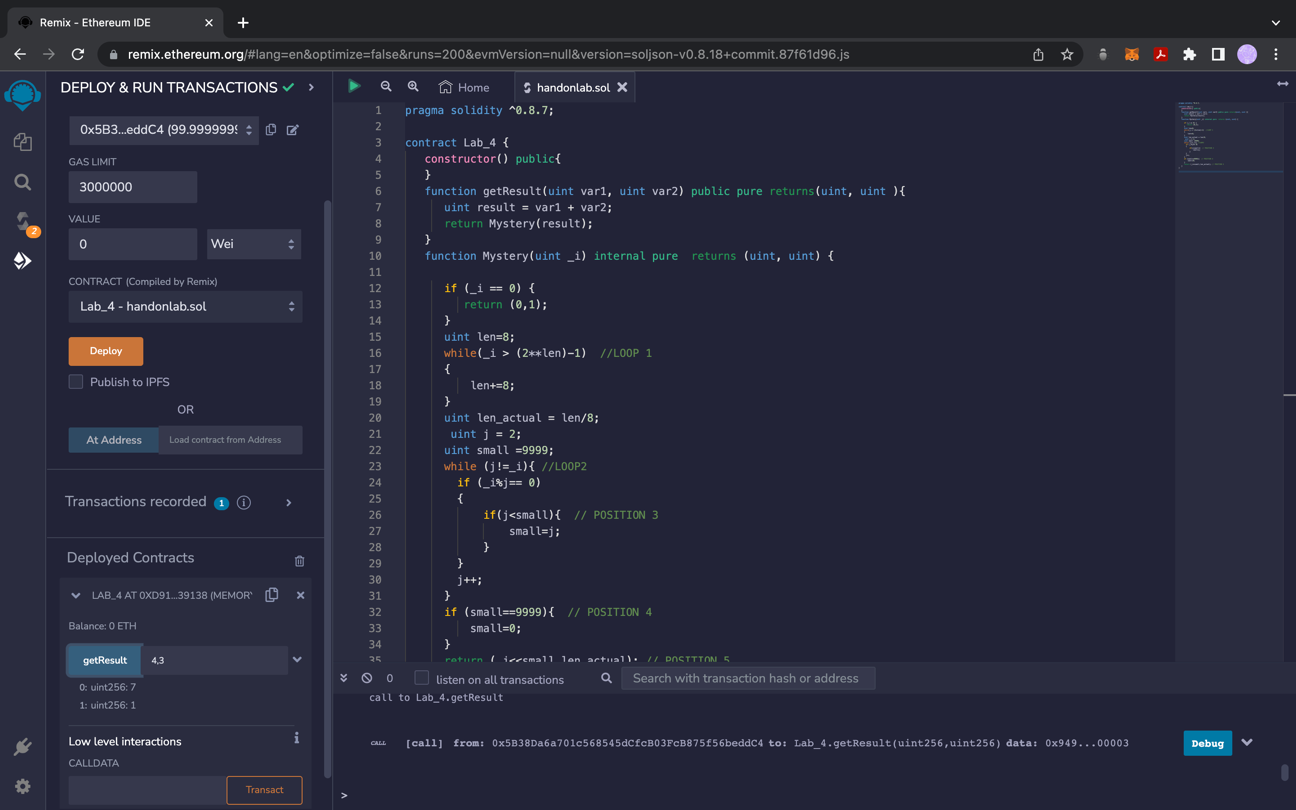
**BLOCKCHAIN (UE20CS335)**

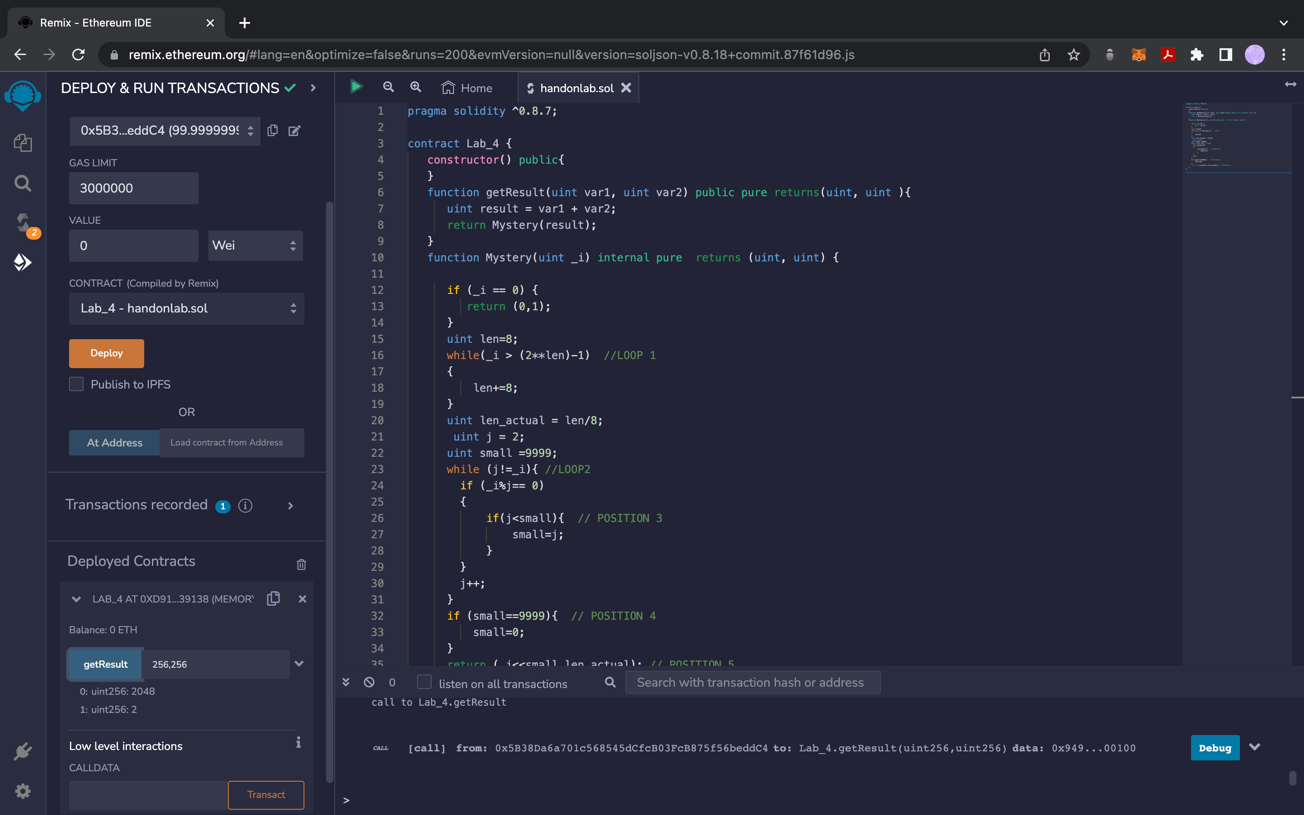
|  |
| --- |
| Name: Vanshika Goel |
| SRN: PES1UG20CS484 |
| Hands On 4 and 5 |

**Task 2: Understanding the code flow**

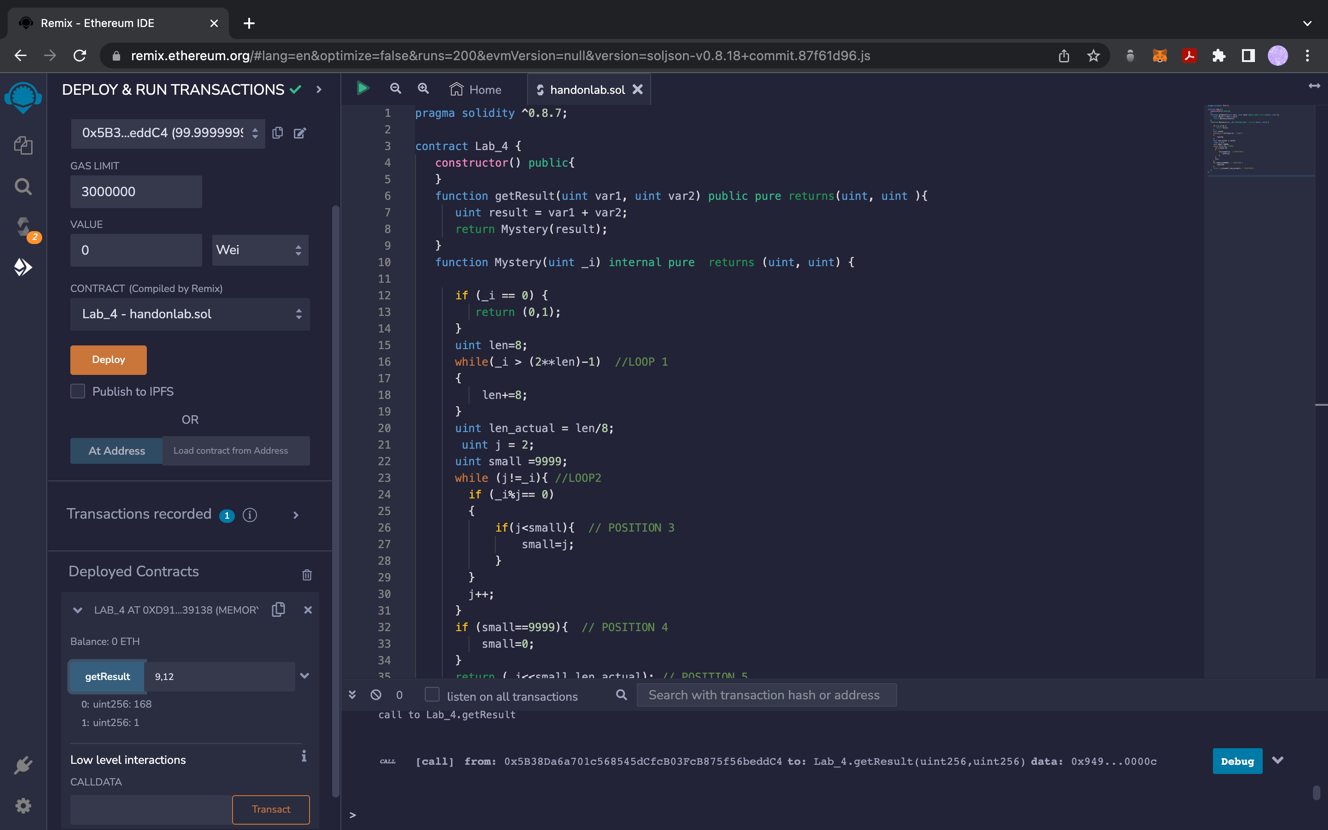
1. Execute the following program and attach screenshots of the output generated for the following test cases:
   1. 4 and 3



* 1. 256 and 256



* 1. 9 and 12



* State what the given program is doing by explaining the outputs generated in each test case.

For the test cases

1. 4, 3

The getResult function calculates the result value = 7 and inputs that into the Mystery function.

The Mystery function enters loop1 to set len to 8 and then goes to loop 2, where we find the smallest factor of 7. Since 7 is a prime number, small is set to 9999.

Thus, the final result is 7 and 1 since 7 is left shift by 0 bits and minimum number of bytes needed are 1.

1. 256, 256

The getResult function calculates the result value = 512 and inputs that into the Mystery function.

The Mystery function enters loop1 to set len to 16, since we need 2 bytes to represent 512 and then goes to loop 2, where we find the smallest factor of 512. The value of small is set to 2 (value of j) because 512 = 2^9.

Thus, the final result is 2048 and 2 since 512 is left shift by 2 bits and len\_actual is 2.

1. 9,12

The getResult function calculates the result value = 21 and inputs that into the Mystery function.

The Mystery function enters loop1 to set len to 8, since we need 1 byte to represent 21 and then goes to loop 2, where we find the smallest factor of 21. The value of small is set to 3 because 21 is divisible by 3.

Thus, the final result is 168 and 1 since 21 is left shift by 3 bits and len\_actual is 1.

* Explain what **Loop1** and **Loop2** does in the given code.
  + Loop 1 checks the minimum number of bytes needed to represent the value of result. Loop 2 find the smallest factor of the value of result variable.
* Replace the statement in **Position 3** with a single line statement that does the same role/task as the statement(/s) given.
  + small = (j<small) ? j: small;
* What does **small** being assigned to 0 in the if condition at **Position 4** indicate?
  + This condition indicates that the value of result variable is a prime number, and that no smaller factor was found for it.
* Make appropriate changes to the **Mystery function** to return the two values without making use of a return statement. Attach a screenshot of the updated function (including the replacement line at Position 3).

// SPDX-License-Identifier: GPL-3.0

pragma solidity ^0.8.7;

contract Lab\_4 {

uint public final\_value;

uint public length;

function Mystery(uint \_i) internal{

if (\_i == 0) {

length = 0;

final\_value= 1;

}

uint len = 8;

while (\_i > (2 \*\* len) - 1) {

len += 8;

}

uint len\_actual = len / 8;

uint j = 2;

uint small = 9999;

while (j != \_i) {

if (\_i % j == 0) {

if (j < small) {

small = (\_i % j == 0 && j < small) ? j : small;

}

}

j++;

}

if (small == 9999) {

small = 0;

}

final\_value = \_i << small;

length = len\_actual;//POSITION 5

}

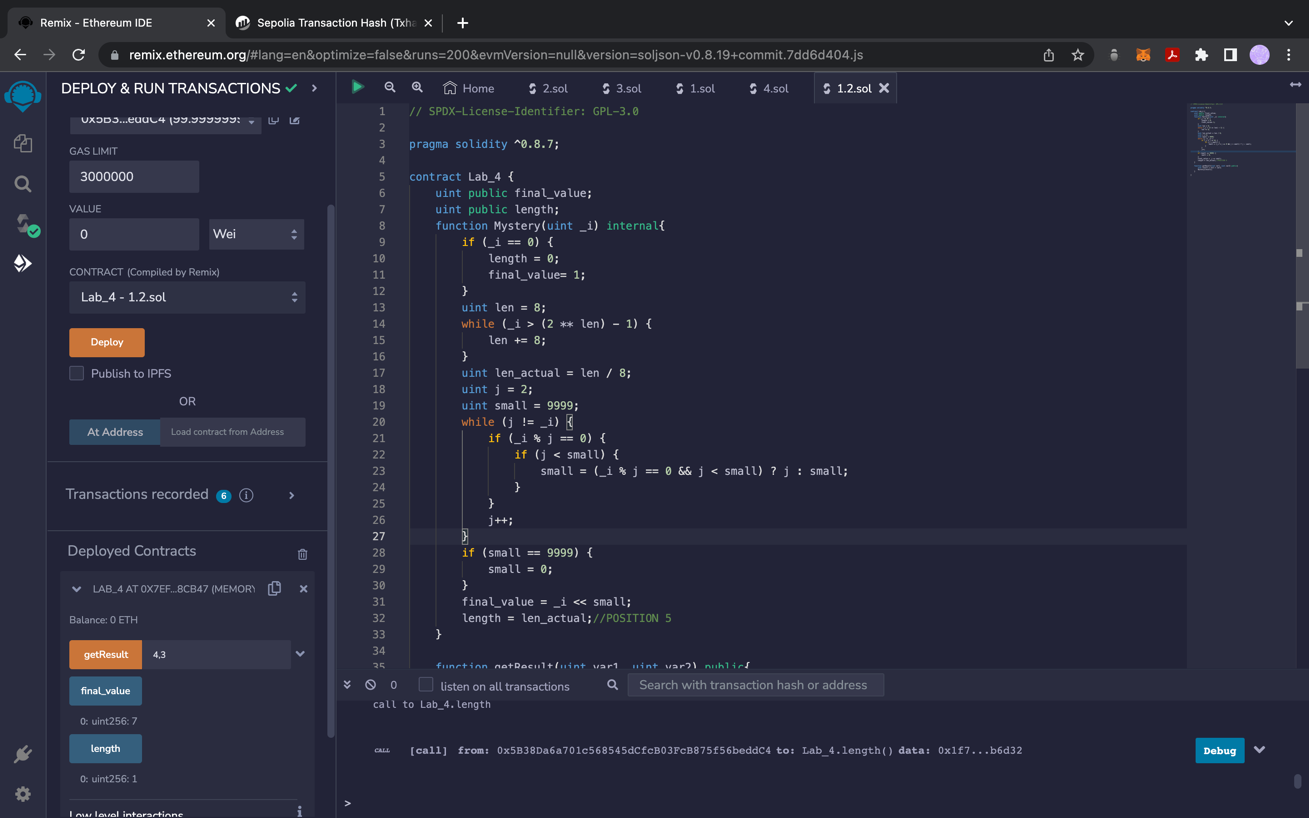
function getResult(uint var1, uint var2) public{

uint result = var1 + var2;

Mystery(result);

}

}



1. Write the Solidity code for a contract **Contract\_XYZ** that follows the given outline-

**Set\_Method(**) function takes the size of the array (**a1**)  and the multiplication factor(**a\_m**) from the user. The function returns the multiplication table of a\_m till size a1.

Example:

**Input**: 5(**a1**), 15(**a\_m**)

**Output**: 0,15,30,45,60

Attach appropriate code and output screenshots in the document before submitting.

pragma solidity ^0.8.0;

contract Contract\_XYZ {

function Set\_Method(uint a1, uint a\_m) public pure returns (uint[] memory) {

uint[] memory result = new uint[](a1);

for (uint i = 0; i < a1; i++) {

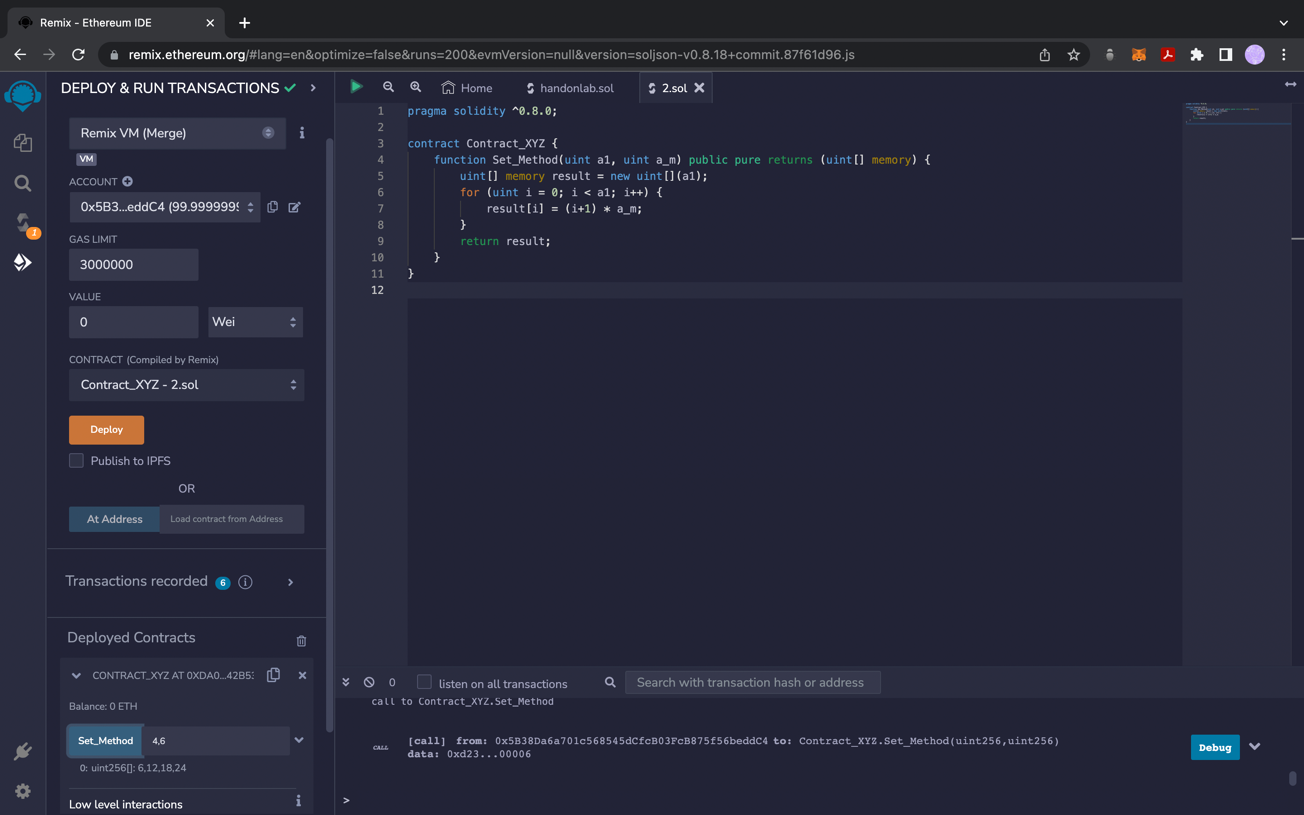
result[i] = (i+1) \* a\_m;

}

return result;

}

}



1. Analyse the given codes on REMIX platform Metamask environment variable.

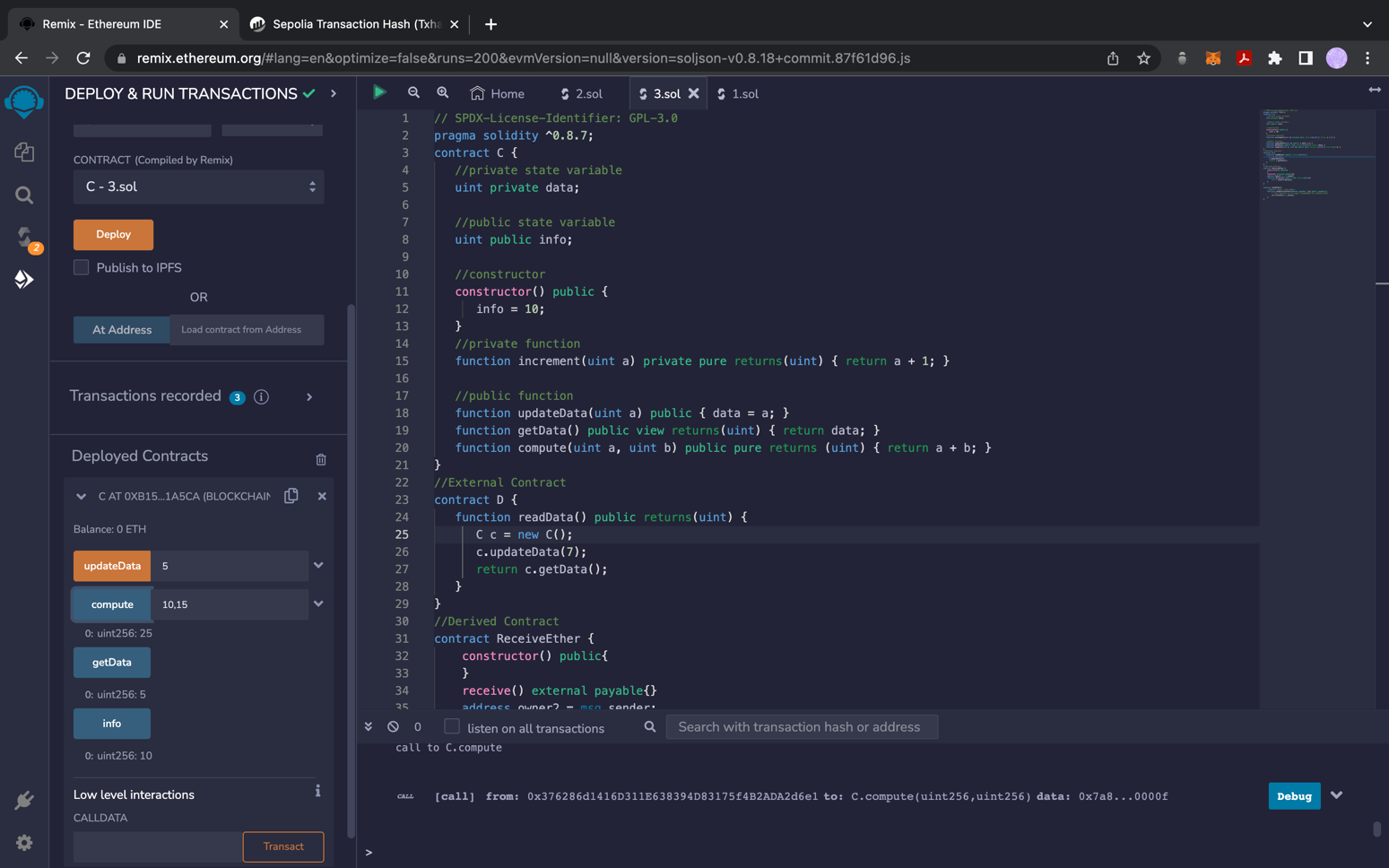
Take appropriate screenshots of the outputs and meta mask wallets after deploying the smart contracts.

Code 1

Graphical user interface, text, application

Description automatically generatedGraphical user interface, text, application

Description automatically generated



Code 2

